Nearly Linear-Time Model-Based Compressive Sensing

Citation: Hegde, Chinmay, et al. “Nearly Linear-Time Model-Based Compressive Sensing.” Automata, Languages, and Programming, edited by Javier Esparza et al., vol. 8572, Springer Berlin Heidelberg, 2014, pp. 588–99.

As Published: http://dx.doi.org/10.1007/978-3-662-43948-7_49

Publisher: Springer Berlin Heidelberg

Persistent URL: http://hdl.handle.net/1721.1/113095

Version: Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

Nearly Linear-Time Model-Based Compressive Sensing

Chinmay Hegde, Piotr Indyk, and Ludwig Schmidt

Massachusetts Institute of Technology, Cambridge MA 02139, USA

Abstract. Compressive sensing is a method for recording a k-sparse signal $x \in \mathbb{R}^n$ with (possibly noisy) linear measurements of the form $y = Ax$, where $A \in \mathbb{R}^{m \times n}$ describes the measurement process. Seminal results in compressive sensing show that it is possible to recover the signal x from $m = O(k \log \frac{n}{k})$ measurements and that this is tight. The *model-based compressive sensing* framework overcomes this lower bound and reduces the number of measurements further to $m = O(k)$. This improvement is achieved by limiting the supports of x to a *structured sparsity model*, which is a subset of all $\binom{n}{k}$ possible k-sparse supports. This approach has led to measurement-efficient recovery schemes for a variety of signal models, including tree-sparsity and block-sparsity.

While model-based compressive sensing succeeds in reducing the number of measurements, the framework entails a computationally expensive recovery process. In particular, two main barriers arise: (i) Existing recovery algorithms involve several *projections* into the structured sparsity model. For several sparsity models (such as tree-sparsity), the best known model-projection algorithms run in time $\Omega(kn)$, which can be too slow for large k. (ii) Existing recovery algorithms involve several matrix-vector multiplications with the measurement matrix A. Unfortunately, the only known measurement matrices suitable for model-based compressive sensing require $O(nk)$ time for a single multiplication, which can be (again) too slow for large k.

In this paper, we remove both aforementioned barriers for two popular sparsity models and reduce the complexity of recovery to *nearly linear time*. Our main algorithmic result concerns the tree-sparsity model, for which we solve the model-projection problem in $O(n \log n + k \log^2 n)$ time. We also construct a measurement matrix for model-based compressive sensing with matrix-vector multiplication in $O(n \log n)$ time for $k \leq n^{1/2-\mu}$, $\mu > 0$. As an added bonus, the same matrix construction can also be used to give a fast recovery scheme for the block-sparsity model.

Keywords: Model-based compressive sensing, model-projection, tree-sparsity, restricted isometry property, compressive sensing.

This work was supported by grants from the MITEI-Shell program, the MADALGO center, and the Packard Foundation. A full version of this paper (including the appendix) is available at `http://people.csail.mit.edu/ludwigs/papers/icalp14_fastmodelcs.pdf`.

1 Introduction

Compressive sensing is a method for recording a signal while taking only a small number of measurements. In particular, recording with *linear measurements* has attracted significant attention over the last decade [CRT06a,Don06,FR13]. In this setup, we are interested in recovering a vector $x \in \mathbb{R}^n$ (the signal) from measurements of the form $y = Ax$, where A is an $m \times n$ matrix and $y \in \mathbb{R}^m$. Usually, the setup also encompasses measurements corrupted by a noise vector e (i.e., $y = Ax+e$), in which case we are interested in recovering a good approximation to x. The main questions in compressive sensing deal with the conditions on A and x that enable efficient, stable recovery from only $m \ll n$ measurements. Compressive sensing has found applications in a wide variety of signal acquisition settings (e.g., MRI [LDP07]) and the underlying problem of *sparse recovery* has connections to several other fields such as data stream algorithms [Mut05,GI10] and Fourier sampling [HIKP12].

Seminal results in compressive sensing show that it is possible to recover a *k-sparse* signal x (containing at most k non-zeros) from $m = O(k \log n/k)$ linear measurements, as long as the measurement matrix A is chosen to satisfy the *restricted isometry property* (RIP) [CRT06b]. Moreover, the recovery step can be performed in polynomial time using several algorithms such as ℓ_1-minimization or CoSaMP [CRT06b,NT09]. While the bound on the number of measurements m is asymptotically tight in the noisy k-sparse setting [DBIPW10,FPRU10], there are ways to overcome this barrier and improve the "compression rate" even further. One such approach for reducing the number of measurements is *model-based compressive sensing* [BCDH10]. In this framework, we make additional assumptions about the support of the signal x. Instead of considering *all* k-sparse signals, we limit our attention to a smaller family of k-sparse supports, which we call a *structured sparsity model* $\mathbb{M}_k$. Research in signal processing has shown that this often is a useful way to capture additional structure in the signals of interest. For example, for some classes of time-domain signals x, the large coefficients in x tend to occur consecutively as clusters. For several sparsity models, it is possible to show measurement bounds of $m = O(k)$. Note that this improvement is not only of theoretical interest: for large values of n, removing the logarithmic factor in m can decrease the measurement complexity by up to an order of magnitude in practice.

While model-based compressive sensing succeeds in reducing the number of measurements, the current framework also entails a computationally more expensive recovery process. In particular, two main barriers limit the recovery performance of model-based compressive sensing compared to "standard" k-sparse compressive sensing:

1. Recovery algorithms for model-based compressive sensing rely on the availability of a *model-projection algorithm*. Given an arbitrary signal x, a model-projection algorithm returns the best approximation of x in the sparsity model $\mathbb{M}_k$. Unfortunately, for many sparsity models, the running time of the best known model-projection algorithm is $\Omega(nk)$.

2. For standard compressive sensing, researchers have identified several classes of measurement matrices A that satisfy the RIP and allow fast matrix-vector multiplication in time $O(n \log n)$; see [NPW14] and references therein. In contrast, matrices known to satisfy the model-equivalent of the RIP only admit slow multiplication in time $O(nm)$ [BCDH10]. Since known recovery algorithms for model-based compressive sensing perform several matrix-vector multiplications, this can become a bottleneck in the overall time complexity. One approach to overcome this barrier is to use *sparse* matrices that satisfy the ℓ_1-variant of the RIP. However, recent work shows that this implies a lower bound of $m = \Omega(k \log \frac{n}{k} / \log \log \frac{n}{k})$ for the tree-sparsity model [IR13].

In this paper, we remove both aforementioned barriers for two popular sparsity models and bring the recovery performance of these models down to *nearly linear time*. Our central results concern the *tree-sparsity model*. In this model, the coefficients of the signal x are arranged as a complete d-ary tree. The model then requires that the support of x forms a connected subtree containing the root node. The tree-sparsity model captures structure in the wavelet-domain representation of natural images; see [Bar99] and [HIS14c] for more details.

As a bonus, our techniques also imply a fast recovery scheme for the *block-sparsity model*. In the block-sparsity model, the signal is divided into a fixed number of blocks, and valid supports can be described as the union of a small number of such blocks. The block-sparsity model captures signal structure in settings where the nonzeros form a small number of clusters.

Our contributions This paper contains two results:

1. Our main technical contribution is a fast model-projection for the tree-sparsity model with time complexity $O(n \log n + k \log^2 n)$. Our formal recovery guarantees complement recent empirical results in [HIS14c].
2. Building on [NPW14], we construct a measurement matrix which satisfies the model-RIP and enables multiplication in $O(n \log n + k^2 \log n \log^2 (k \log n))$ time for general k. For $k \leq n^{1/2-\mu}$, $\mu > 0$, the multiplication time is $O(n \log n)$. Moreover, our matrix has the same bound on the number of measurements as existing, slow model-RIP matrices: $m = O(k + \log|\mathbb{M}_k|)$.

Together with existing results [BCDH10,HIS14b], our contributions enable us to state recovery guarantees of the following form: Let x be a signal in the tree-sparsity model with sparsity parameter k and let A be our new measurement matrix with $m = O(k)$ rows. The measurements are given by $y = Ax + e$ for arbitrary noise e. Then we can recover an $\widehat{x}$ such that $\|x - \widehat{x}\|_2 \leq C \|e\|_2$. Moreover, we can perform the recovery in time $O((n \log n + k^2 \log n \log^2 (k \log n)) \log \frac{\|x\|_2}{\|e\|_2})$. Note that this compares favorably with the time complexity of the original model-based compressive sensing framework [BCDH10]: $O(nk \log \frac{\|x\|_2}{\|e\|_2})$. Table 1 compares our results to previous recovery schemes for the tree-sparsity model. Our recovery guarantees for the block-sparsity model are analogous.

Ideally, a model-RIP matrix with $m = O(k + \log|\mathbb{M}_k|)$ rows would offer a multiplication time of $O(n \log n)$ for all values of k. However, we conjecture that such a result is connected to progress on the measurement bound for subsampled

Paper	Measurement bound	Recovery time	Matrix-vector multiplication time	Recovery guarantee
[BCDH10]	$O(k)$	$O(nk)$	$O(nk)$	ℓ_2
[IR13]	$O\big(k\,\frac{\log n}{\log\log n}\big)$	exponential	$O(n\log n)$	ℓ_1
[BBC14]	$O\big(k\,\frac{\log n}{\log\log n}\big)$	$O(nk)$	$O(n\log n)$	ℓ_1
This paper	$O(k)$	$O(n\log n)$	$O(n\log n)$	ℓ_2

Table 1. Comparison of our results with previous recovery schemes for the tree-sparsity model. In order to simplify the presentation, all stated bounds are for the regime of $k \le n^{1/2-\mu}$ with $\mu > 0$. We also omit a factor of $\log\frac{\|x\|_2}{\|e\|_2}$ from all recovery times. An ℓ_p-recovery guarantee is of the form $\|x-\widehat{x}\|_p \le C\|e\|_p$, where x is the original signal, $\widehat{x}$ is the recovery result, e is the measurement noise, and C is a fixed constant.

Fourier matrices in k-sparse compressive sensing. This is considered a challenging open problem in the field.

Our techniques We achieve the aforementioned results with the following tools:

1. In order to project into the tree-sparsity model, we use the recent framework for *approximation-tolerant* model-based compressive sensing [HIS14b], which was originally introduced for another sparsity model. Following this framework, instead of providing a single *exact* model-projection algorithm, we give two *approximate* algorithms: one for the *minimization* and one for the *maximization* version of the problem. The first algorithm builds a solution by combining several small subtrees which are cheap to find. The second algorithm works with a Lagrangian relaxation and constructs the corresponding Pareto curve with a sweep line approach.
2. We construct our measurement matrix by combining a fast standard-RIP matrix for initial dimensionality reduction with a standard model-RIP matrix for achieving a small number of measurements.

Related work There is a large body of work on matrices satisfying the RIP for general k-sparse vectors (e.g. see [RV08,BDDW08,GI10,CGV13] and references therein). For matrices with fast matrix-vector multiplication in $O(n\log n)$ time, the best known measurement bound is $m = O(k\log n\log^2(k\log n))$ [NPW14]. For $k \le n^{1/2-\mu}$ and $\mu > 0$, there exist fast matrices with $m = O(k\log n)$ [AR13]. Note that in this regime, $O(k\log n) = O(k\log\frac{n}{k})$.

For the model-RIP, the only known matrices with $m = O(k + \log|\mathbb{M}_k|)$ are *dense* matrices with i.i.d. subgaussian entries [BCDH10]. Vector-matrix multiplication with such matrices requires $O(mn)$ time. While ℓ_1-model-RIP matrices support faster multiplication, they also entail a measurement lower bound of $m = \Omega(k\log\frac{n}{k}/\log\log\frac{n}{k})$ for the tree-sparsity model [IR13].

The problem of projecting into the tree-sparsity model has received a fair amount of attention in the literature over the last two decades. Researchers have proposed several algorithms such as the condensing sort-and-select algorithm (CSSA) [BJ94], complexity-penalized residual sum-of-squares (CPRSS) [Don97], and optimal pruning [BB94]. However, all of these algorithms either run in time

$\Omega(n^2)$ or fail to provide projection guarantees for general input signals. A recent paper describes a dynamic programming algorithm for exact projections running in time $O(nk)$ [CT13]. Combining this algorithm with the ℓ_1-model-RIP matrices mentioned above, another recent paper provides a compressive sensing recovery scheme in the ℓ_1-setting [BBC14]. As a result, the measurement complexity is constrained by the aforementioned lower bound and the recovery time is $\Omega(nk)$.

In related work, an algorithm for *approximate* projections into the tree-sparsity model has been proposed [HIS14c]. Unfortunately, this algorithm only has a weakly polynomial running time depending on the largest and smallest nonzero absolute values in the input. Moreover, it solves only the minimization variant of the problem, which is not sufficient to establish a compressive sensing recovery result. Instead, the authors demonstrate the validity of their approach via several numerical experiments. Our results here complement these findings with formal guarantees. We note that our minimization algorithm is related to the algorithm in [HIS14c] but achieves a strongly polynomial running time.

2 Preliminaries

Structured sparsity A signal $x \in \mathbb{R}^n$ is k-sparse if at most k of its coefficients are nonzero. The support of x, denoted by $\text{supp}(x) \subseteq [n]$, contains the indices corresponding to the nonzero entries in x.

Suppose that we posses some additional information about the support of our signals of interest. One way to model this information is as follows [BCDH10]: denote the set of *allowed supports* with $\mathbb{M}_k = \{\Omega_1, \Omega_2, \ldots, \Omega_L\}$, where $\Omega_i \subseteq [n]$ and $|\Omega_i| = k$. Often it is useful to work with the closure of $\mathbb{M}_k$ under taking subsets, which we denote with $\mathbb{M}_k^+ = \{\Omega \subseteq [n] \mid \Omega \subseteq S \text{ for some } S \in \mathbb{M}_k\}$. Then, we define a *structured sparsity model*, $\mathcal{M}_k \subseteq \mathbb{R}^n$, as the set of vectors such that $\mathcal{M}_k = \{x \in \mathbb{R}^n \mid \text{supp}(x) \in \mathbb{M}_k^+\}$. The number of allowed supports $L = |\mathbb{M}_k|$ is called the "size" of the model $\mathcal{M}_k$; typically $|\mathbb{M}_k| \ll \binom{n}{k}$.

Our central focus in this paper is the *tree-sparsity model* [BCDH10]. Let n be such that the coefficients of a signal $x \in \mathbb{R}^n$ can be arranged as the nodes of a perfect d-ary tree rooted at node 1.[1] Then, the tree-sparsity model comprises the set of k-sparse signals whose nonzero coefficients form a *connected subtree* rooted at node 1. More formally, let $\mathbb{T}$ be the set of supports forming a connected subtree and let $\mathbb{T}_i$ be the set of supports forming a connected subtree rooted at node i. Then the tree-sparsity model is defined as $\mathbb{M}_k = \{\Omega \subseteq [n] \mid \Omega \in \mathbb{T}_1 \text{ and } |\Omega| = k\}$. The size of this model is bounded by $|\mathbb{M}_k| \leq (2e)^k/(k+1)$ [BCDH10]. For a subtree Ω with root r, we use root-path(Ω) to denote the set of nodes on the path from r to node 1 (the root of the entire tree).

Model projections For a sparsity model $\mathcal{M}_k$, we define the problem of *model-projection* as follows: given $x \in \mathbb{R}^n$, find a $x^* \in \mathcal{M}_k$ such that $\|x - x^*\|_p$ is minimized for a norm parameter $p \geq 1$. In general, this problem can be hard

[1] Our algorithms can easily be extended to handle *complete* d-ary trees and hence work for the general tree-sparsity model with arbitrary dimension n. For simplicity, we state our algorithms here for the special case of *perfect* d-ary trees.

since $\mathcal{M}_k$ is typically non-convex. Moreover, the original model-based compressive sensing framework in [BCDH10] requires the minimization to be *exact*. An alternative is the *approximation-tolerant* model-based compressive sensing framework [HIS14b]. Instead of a single *exact* model-projection algorithm, the framework requires two *approximate* model-projection algorithms with two different notions of approximation:

- A head approximation algorithm $H(x, k)$ that satisfies the following guarantee: Let $\widehat{\Omega} = H(x, k)$. Then $\widehat{\Omega} \in \mathbb{M}^+_{c_1 k}$ and $\|x_{\widehat{\Omega}}\|_p \geq c_2 \max_{\Omega \in \mathbb{M}_k} \|x_\Omega\|_p$ for some constants $c_1 \geq 1$ and $c_2 \leq 1$.
- A tail approximation algorithm $T(x, k)$ that satisfies the following guarantee: Let $\widehat{\Omega} = T(x, k)$. Then $\widehat{\Omega} \in \mathbb{M}^+_{c_1 k}$ and $\|x - x_{\widehat{\Omega}}\|_p \leq c_2 \min_{\Omega \in \mathbb{M}_k} \|x - x_\Omega\|_p$ for some constants $c_1 \geq 1$ and $c_2 \geq 1$.

Using such approximate model-projection algorithms, the framework of [HIS14b] provides the same asymptotic recovery guarantees as those achieved with an exact model-projection.

Measurement matrices Many recovery algorithms for compressive sensing assume that the measurement matrix satisfies the *restricted isometry property* (RIP). A matrix $A \in \mathbb{R}^{m \times n}$ has the (δ, k)-RIP if the following inequalities hold for all k-sparse vectors $x \in \mathbb{R}^n$:

$$(1 - \delta)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1 + \delta)\|x\|_2^2 . \tag{1}$$

There exist measurement matrices satisfying the RIP with only $m = O(k \log \frac{n}{k})$ rows [BDDW08]. A matrix $A \in \mathbb{R}^{m \times n}$ has the (δ, k)-*model-RIP* for model $\mathcal{M}_k$ if (1) holds for all $x \in \mathcal{M}_k$. There exist matrices satisfying the model-RIP with only $m = O(k + \log|\mathbb{M}_k|)$ rows [BCDH10].

Recovery algorithm We briefly summarize the *approximate model-iterative hard thresholding* (AM-IHT) algorithm for signal recovery using approximate model projections. For a full explanation, see [HIS14b] and references therein. Let $y = Ax + e$, where e is the measurement noise vector. Then, one can recover a signal estimate $\widehat{x}$ satisfying $\|x - \widehat{x}\|_2 \leq C\|e\|_2$ by applying the following update rule, inspired by the well-known iterative hard thresholding (IHT) [BD09]:

$$x^{(i+1)} \leftarrow T(x^{(i)} + H(A^T(y - Ax^{(i)}))) . \tag{2}$$

It is possible to show that $O(\log \frac{\|x\|_2}{\|e\|_2})$ iterations suffice for guaranteed recovery. Therefore, the overall time complexity of AM-IHT is governed by the running times of $H(\cdot)$, $T(\cdot)$, and the cost of matrix-vector multiplication with A and A^T.

3 Head approximation for the tree-sparsity model

We propose a head approximation algorithm for the tree-sparsity model. In order to simplify the analysis, we will assume that $k \geq \lceil \log_d n \rceil$. Note that we can always reduce the input to this case by removing layers of the tree with depth greater than k. Our approach is based on the following structural result about decompositions of d-ary trees, which we prove in Appendix A.

Algorithm 1 (HeadApprox) Head approximation for the tree-sparsity model

```
1: function HeadApprox$(x, k, d, p, \alpha)$
2:     Run ETP on $x$ with sparsity parameter $k' = d\alpha$.
3:     $x^{(1)} \leftarrow x$
4:     for $i \leftarrow 1, \ldots, \lceil \frac{k}{\alpha} \rceil$ do
5:         $\widehat{\Omega}_i \leftarrow \operatorname*{arg\,max}_{\Omega \in \mathbb{T},\, |\Omega| = d\alpha} \|x^{(i)}_{\Omega}\|_p$
6:         $x^{(i+1)} \leftarrow x^{(i)}, \qquad x^{(i+1)}_{\widehat{\Omega}_i} \leftarrow 0$
7:         for $j \in \widehat{\Omega}_i \cup$ root-path$(\widehat{\Omega}_i)$ (in bottom-up order) do
8:             Update the DP table for node $j$ up to sparsity $k' = d\alpha$.
9:     return $\widehat{\Omega} \leftarrow \bigcup_{i=1}^{\lceil \frac{k}{\alpha} \rceil} \widehat{\Omega}_i \cup$ root-path$(\widehat{\Omega}_i)$
```

Lemma 1. *Let T be a d-ary tree with $|T| = k$. Moreover, let $\alpha \in \mathbb{N}, \alpha \geq 1$. Then T can be decomposed into a set of disjoint, connected subtrees $S = \{T_1, \ldots, T_\beta\}$ such that $|T_i| \leq d\alpha$ for all $i \in [\beta]$ and $\beta = |S| \leq \lceil \frac{k}{\alpha} \rceil$.*

In addition to the tree decomposition, our head-approximation algorithm builds on the exact tree projection algorithm (ETP) introduced in [CT13]. The algorithm finds the best tree-sparse approximation for a given signal via dynamic programming (DP) in $O(nkd)$ time.[2] We run ETP with a small sparsity value $k' < k$ in order to find optimal subtrees of size k'. We then assemble several such subtrees into a solution with a provable approximation guarantee. We use the fact that ETP calculates the DP table entries in the following way: if the DP tables corresponding to the children of node i are correct, the DP table for node i can be computed in $O(k'^2)$ time. The time complexity follows from the structure of the DP tables: for every node and $l \leq k'$, we store the value of the best subtree achievable at that node with sparsity exactly l. We can now state our head-approximation algorithm (Alg. 1) and the corresponding guarantees.

Theorem 1. *Let $x \in \mathbb{R}^n$ be the coefficients corresponding to a d-ary tree rooted at node 1. Also, let $p \geq 1$ and $\alpha \geq 1$. Then* HeadApprox(x, k, d, p, α) *returns a support $\widehat{\Omega}$ satisfying $\left\|x_{\widehat{\Omega}}\right\|_p \geq \left(\frac{1}{4}\right)^{1/p} \max_{\Omega \in \mathbb{M}_k} \|x_\Omega\|_p$. Moreover, $\widehat{\Omega} \in \mathbb{M}^+_\gamma$ with $\gamma = \lceil \frac{k}{\alpha} \rceil (d\alpha + \lceil \log_d n \rceil)$.*

Proof. Let $\Omega^* \in \mathbb{M}_k$ be an optimal support, i.e., $\|x_{\Omega^*}\|_p = \max_{\Omega \in \mathbb{M}_k} \|x_\Omega\|_p$. Using Lemma 1, there is a decomposition of Ω^* into disjoint sets $\Omega^*_1, \ldots, \Omega^*_\beta$ such that $\Omega^*_i \in \mathbb{T}$, $|\Omega^*_i| \leq d\alpha$ and $\beta \leq \lceil \frac{k}{\alpha} \rceil$. The contribution of Ω^*_i to the overall solution is $\|x_{\Omega^*_i}\|^p_p$. Now, compare the contributions of our subtrees $\widehat{\Omega}_i$ to these quantities. When finding $\widehat{\Omega}_i$ for $i \in [\beta]$, one of the following two cases holds:

1. $\|x^{(i)}_{\Omega^*_i}\|^p_p \geq \frac{1}{2}\|x_{\Omega^*_i}\|^p_p$. Since Ω^*_i is a candidate in the search for $\widehat{\Omega}_i$ in line 5, we have $\|x^{(i)}_{\widehat{\Omega}_i}\|^p_p \geq \|x^{(i)}_{\Omega^*_i}\|^p_p \geq \frac{1}{2}\|x_{\Omega^*_i}\|^p_p$.

[2] While ETP as stated in [CT13] works for $p = 2$ only, the algorithm can easily be extended to arbitrary norm parameters p.

2. $\|x^{(i)}_{\Omega^*_i}\|^p_p < \frac{1}{2}\|x_{\Omega^*_i}\|^p_p$. Therefore, $\widehat{\Omega}_1, \ldots, \widehat{\Omega}_{i-1}$ have already covered at least half of the contribution of Ω^*_i. Formally, let $C_i = \Omega^*_i \cap \bigcup_{j=1}^{i-1} \widehat{\Omega}_j$. Then $\|x_{C_i}\|^p_p \geq \frac{1}{2}\|x_{\Omega^*_i}\|^p_p$.

Let $A = \{i \in [\beta] \,|\, \text{case 1 holds for } \widehat{\Omega}_i\}$ and $B = \{i \in [\beta] \,|\, \text{case 2 holds for } \widehat{\Omega}_i\}$. For the set A we have

$$\|x_{\widehat{\Omega}}\|^p_p \quad = \quad \sum_{i=1}^{\lceil \frac{k}{\alpha} \rceil} \|x^{(i)}_{\widehat{\Omega}_i}\|^p_p \quad \geq \quad \sum_{i \in A} \|x^{(i)}_{\widehat{\Omega}_i}\|^p_p + \sum_{i \in B} \|x^{(i)}_{\widehat{\Omega}_i}\|^p_p \quad \geq \quad \frac{1}{2} \sum_{i \in A} \|x_{\Omega^*_i}\|^p_p \,. \tag{3}$$

Now, consider the set B. Since the Ω^*_i are disjoint, so are the C_i. Moreover, $C_i \subseteq \widehat{\Omega}$ and therefore

$$\|x_{\widehat{\Omega}}\|^p_p \quad \geq \quad \sum_{i=1}^{\beta} \|x_{C_i}\|^p_p \quad \geq \quad \sum_{i \in B} \|x_{C_i}\|^p_p \quad \geq \quad \frac{1}{2} \sum_{i \in B} \|x_{\Omega^*_i}\|^p_p \,. \tag{4}$$

Combining (3) and (4), we get

$$2\|x_{\widehat{\Omega}}\|^p_p \quad \geq \quad \frac{1}{2} \sum_{i \in A} \|x_{\Omega^*_i}\|^p_p + \frac{1}{2} \sum_{i \in B} \|x_{\Omega^*_i}\|^p_p \quad \geq \quad \frac{1}{2} \|x_{\Omega^*}\|^p_p \,.$$

Raising both sides to power $1/p$ gives the guarantee in the theorem. For the sparsity bound, note that $|\widehat{\Omega}_i| \leq d\alpha$ and $|\text{root-path}(\widehat{\Omega}_i)| \leq \lceil \log_d n \rceil$. Since we take the union over $\lceil \frac{k}{\alpha} \rceil$ such sets, the theorem follows. □

We defer the runtime analysis to Appendix A (Theorem 4) and state the final result here. Its proof is a direct consequence of Theorems 1 and 4.

Corollary 1. *Let* $\alpha = \lceil \log_d n \rceil$. *Then* HeadApprox$(x, k, d, p, \alpha)$ *returns a support* $\widehat{\Omega} \in \mathbb{M}^+_{k(2d+2)}$ *satisfying* $\left\|x_{\widehat{\Omega}}\right\|_p \geq \left(\frac{1}{4}\right)^{1/p} \max_{\Omega \in \mathbb{M}_k} \|x_\Omega\|_p$. *Moreover, the algorithm runs in time* $O(n \log n + k \log^2 n)$ *for fixed* d.

4 Tail approximation for the tree-sparsity model

Next, we propose a tail approximation algorithm. We consider the Lagrangian relaxation $\arg\min_{\Omega \in \mathbb{T}_1} \|x - x_\Omega\|^p_p + \lambda|\Omega|$, where the parameter λ controls the trade-off between the approximation error and the sparsity of the identified support. The algorithm in [HIS14c] proceeds by performing a binary search over λ in order to explore the Pareto curve of this trade-off. Unfortunately, the running time of this algorithm is only weakly polynomial because it depends on both $x_{\max} = \max_{i \in [n]} |x_i|$ and $x_{\min} = \min_{i \in [n], |x_i| > 0} |x_i|$. Below, we develop an algorithm that exploits the structure of the Pareto curve in more detail and runs in strongly polynomial time $O(n \log n)$. In fact, our new algorithm constructs the shape of the *entire* Pareto curve and not only a single trade-off.

The Lagrangian relaxation is equivalent to $\arg\max_{\Omega \in \mathbb{T}_1} \|x_\Omega\|^p_p - \lambda|\Omega|$. Hence, we can rewrite this problem as $\arg\max_{\Omega \in \mathbb{T}_1} \sum_{i \in \Omega} y_i$, where $y_i = |x_i|^p - \lambda$. So for a given value of λ, the goal is to find a subtree Ω rooted at node 1 which maximizes the sum of weights y_i associated with the nodes in Ω.

In the following, we analyze how the solution to this problem changes as a function of λ and use this structure in our tail-approximation algorithm. On a high level, the optimal contribution of a node i is positive and decreasing up to a certain value of $\lambda = \gamma_i$, after which the contribution stays 0. So for $\lambda < \gamma_i$, a subtree rooted at node i can contribute positively to an overall solution. For $\lambda \geq \gamma_i$, we can ignore the subtree rooted at node i.

4.1 Properties of the Pareto curve

Let $b_i(\lambda)$ denote the maximum value achievable with a subtree rooted at i:

$$b_i(\lambda) = \max_{\Omega \in \mathbb{T}_i} \|x_\Omega\|_p^p - \lambda|\Omega| \ .$$

Our algorithm relies on two main insights: (i) $b_i(\lambda)$ is a piecewise linear function with at most n non-differentiable points (or "corners"), which correspond to the values of λ at which the optimal support changes. (ii) Starting with $\lambda = 0$, $b_i(\lambda)$ is strictly decreasing up to a certain value of λ, after which $b_i(\lambda) = 0$. Formally, we can state the properties of the Pareto curve as follows.

Lemma 2. *$b_i(\lambda)$ is piecewise linear. There is a value γ_i such that $b_i(\lambda) = 0$ for $\lambda \geq \gamma_i$ and $b_i(\lambda)$ is strictly decreasing for $\lambda \leq \gamma_i$. The corners of $b_i(\lambda)$ are the points $D_i = \{\gamma_i\} \cup \{\gamma \in \bigcup_{j \in children(i)} D_j \mid \gamma < \gamma_i\}$.*

Proof. A simple inductive argument shows that $b_i(\lambda)$ can be recursively defined as

$$b_i(\lambda) = \max(0, \ |x_i|^p - \lambda + \sum_{j \in \text{children}(i)} b_j(\lambda)) \ .$$

Note that the theorem holds for the leaves of the tree. By induction over the tree, we also get the desired properties for all nodes in the tree. We are using the fact that piecewise linear functions and strictly decreasing functions are closed under addition. Moreover, the corners of a sum of piecewise linear functions are contained in the union of the corners of the individual functions. □

Our algorithm does not compute the $b_i(\lambda)$ directly but instead keeps track of the following two quantities $s_i(\lambda)$ and $c_i(\lambda)$. For a given value of λ, $s_i(\lambda)$ denotes the sum achieved by the best subtree rooted at node i. Similarly, $c_i(\lambda)$ denotes the cardinality of the best subtree rooted at node i. These two quantities are easier to maintain algorithmically because they are piecewise constant. The proof of the next lemma follows directly from Lemma 2 and a similar inductive argument. Appendix B.1 contains further properties of the Pareto curve with accompanying proofs.

Lemma 3. *Let*

$$s_i(\lambda) = |x_i|^p + \sum_{\substack{j \in children(i) \\ b_j(\lambda) > 0}} s_j(\lambda) \qquad \textit{and} \qquad c_i(\lambda) = 1 + \sum_{\substack{j \in children(i) \\ b_j(\lambda) > 0}} c_j(\lambda) \ .$$

Then $s_i(\lambda)$ and $c_i(\lambda)$ are piecewise constant and monotonically decreasing. The discontinuities of $s_i(\lambda)$ and $c_i(\lambda)$ are D_i (see Lemma 2). At a discontinuity $\gamma \in D_i$ we have $\lim_{\delta \to 0^+} s_i(\gamma + \delta) = s_i(\gamma)$ and $\lim_{\delta \to 0^+} c_i(\gamma + \delta) = c_i(\gamma)$.

Algorithm 2 (FindPareto) Constructing the Pareto curve

```
function FindPareto($x, p$)
    for $i \leftarrow 1, \ldots, n$ do                                   ▷ Initialization
        $s_i \leftarrow |x_i|^p$,   $c_i \leftarrow 1$,   active$_i \leftarrow$ false
    $\widehat{\lambda}_0 \leftarrow +\infty$
    $r_1 \leftarrow c_1$
    for $i = 1, \ldots, n$ do                          ▷ Iterate over the discontinuities
        $j \leftarrow \operatorname*{arg\,max}_{l \in [n],\ \text{active}_l = \text{false}} \frac{s_l}{c_l}$           ▷ Find the next discontinuity
        $\widehat{\lambda}_i \leftarrow \frac{s_j}{c_j}$
        active$_j \leftarrow$ true
        $a \leftarrow j$
        while $a \neq 1$ do                                 ▷ Update the affected nodes
            $a \leftarrow$ parent($a$)
            $s_a \leftarrow |x_a|^p$
            $c_a \leftarrow 1$
            for $l \in$ children($a$) with active$_l$ = true do
                $s_a \leftarrow s_a + s_l$
                $c_a \leftarrow c_a + c_l$
        $r_{i+1} \leftarrow c_1$
    $\widehat{\lambda}_{n+1} \leftarrow 0$
    return $(\widehat{\lambda}, r)$
```

4.2 Constructing the Pareto curve

We now use the quantities introduced above in order to traverse the Pareto curve. We start with $\lambda = +\infty$, for which the values of the $s_i(\lambda)$ and $c_i(\lambda)$ are easy to determine. Then, we iterate the following two steps (see Algorithm 2): (i) Use the current values of the $s_i(\lambda)$ and $c_i(\lambda)$ to find the next discontinuity. (ii) Update the $s_i(\lambda)$ and $c_i(\lambda)$ based on the change in the optimal support. In order to simplify the analysis, we assume that the discontinuities γ_i are distinct.

Theorem 5 (Appendix B.2) establishes a connection between the variables s_j and c_j in FindPareto and the functions $s_j(\lambda)$ and $c_j(\lambda)$. Using this connection, we can now show that the algorithm returns the shape of the Pareto curve.

Theorem 2. *Let $p \geq 1$ and $x \in \mathbb{R}^n$ and let $\widehat{\lambda}$ and r be the vectors returned by* FindPareto(x, p). *Moreover, let $\lambda > 0$ such that $\widehat{\lambda}_{i-1} > \lambda \geq \widehat{\lambda}_i$. Then we have $r_i = |\Omega^*_\lambda|$ where*

$$\Omega^*_\lambda = \operatorname*{arg\,max}_{\substack{\Omega \in \mathbb{T}_1,\ 1 \in \Omega \\ b_j(\lambda) > 0 \ for \ j \in \Omega \setminus \{1\}}} \|x_\Omega\|_p^p - \lambda|\Omega| \ .$$

Proof. By the definition of FindPath and Theorem 5, we have $r_i = c_1(\lambda)$ for $\widehat{\lambda}_{i-1} > \lambda \geq \widehat{\lambda}_i$. The theorem then follows from Lemma 4 (Appendix B.1). □

Moreover, FindPareto can be implemented to run in $O(n \log n)$ time using a priority queue; see Theorem 6 in Appendix B.2 for a formal runtime analysis.

Given the shape of the Pareto curve, we can traverse it to find a suitable trade-off parameter $\widehat{\lambda}$ that achieves a constant-factor tail approximation. The main idea of this last claim is similar to the algorithm in [HIS14c]; we state the final guarantee with proof and pseudo code in Appendix B.3.

5 Compressive Sensing Recovery

We have developed constant factor head and tail approximation algorithms for the tree-sparsity model, both of which run in near-linear time $O(n \log n)$. Therefore, we can invoke AM-IHT (Eq. (2)) to achieve an algorithm for recovering tree-sparse signals from (noisy) linear measurements.[3] In Appendix C, we describe a new construction of a matrix $A \in \mathbb{R}^{m \times n}$ that satisfies the model-RIP for the tree-sparsity model $\mathcal{M}_k$ and in addition supports fast matrix-vector multiplication. Combining these ingredients, we obtain:

Theorem 3. *Let $A \in \mathbb{R}^{m \times n}$ be a model-RIP matrix as constructed in the proof of Theorem 8. Let $x \in \mathbb{R}^n$ be a signal with $x \in \mathcal{M}_k$ and let $y = Ax + e$ be the noisy measurements. Then, there exists an algorithm to recover a signal estimate $\widehat{x} \in \mathcal{M}_{ck}$ from y such that $\|x - \widehat{x}\|_2 \leq C\|e\|_2$ for some constants $c > 1$, $C > 0$. The algorithm runs in $O((n \log n + k^2 \log n \log^2(k \log n)) \log \frac{\|x\|_2}{\|e\|_2})$ time for general k, and in $O(n \log n)$ time for the range $k \leq n^{1/2-\mu}$ with $\mu > 0$.*

While we have stated our results for the tree-sparsity model, a completely analogous construction of A with optimal parameters is possible in the context of the *block-sparsity* model of [BCDH10]. In particular, since the block-sparse projection can be computed exactly in linear time, this construction yields near-linear time recovery of block-sparse signals. We omit a detailed derivation.

References

AR13. N. Ailon and H. Rauhut. Fast and RIP-optimal transforms. *Preprint*, 2013. `http://arxiv.org/abs/1301.0878`.

Bar99. R. Baraniuk. Optimal tree approximation with wavelets. In *SPIE Wavelet Applications in Signal and Image Processing*, 1999.

BB94. M. Bohanec and I. Bratko. Trading accuracy for simplicity in decision trees. *Machine Learning*, 1994.

BBC14. B. Bah, L. Baldassarre, and V. Cevher. Model-based sketching and recovery with expanders. In *Symposium on Discrete Algorithms (SODA)*, 2014.

BCDH10. R. Baraniuk, V. Cevher, M. Duarte, and C. Hegde. Model-based compressive sensing. *IEEE Trans. Inform. Theory*, 2010.

BD09. T. Blumensath and M. Davies. Iterative hard thresholding for compressed sensing. *Appl. Comput. Harmon. Anal.*, 2009.

BDDW08. R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 2008.

BJ94. R. Baraniuk and D. Jones. A signal-dependent time-frequency representation: Fast algorithm for optimal kernel design. *IEEE Trans. Sig. Proc.*, 1994.

[3] To be precise, the AM-IHT algorithm proposed in [HIS14b] imposes additional restrictions on the approximation factors of the head and tail algorithms. However, it is possible to modify AM-IHT to work with arbitrary constant factors. See [HIS14a], which is the journal version of [HIS14b].

CGV13. M. Cheraghchi, V. Guruswami, and A. Velingker. Restricted isometry of Fourier matrices and list decodability of random linear codes. In *Symposium on Discrete Algorithms (SODA)*, 2013.

CRT06a. E. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inform. Theory*, 2006.

CRT06b. E. Candes, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Comm. Pure Appl. Math.*, 2006.

CT13. C. Cartis and A. Thompson. An exact tree projection algorithm for wavelets. *IEEE Signal Process. Lett.*, 2013.

DBIPW10. K. Do Ba, P. Indyk, E. Price, and D. Woodruff. Lower bounds for sparse recovery. In *Symposium on Discrete Algorithms (SODA)*, 2010.

Don97. D. Donoho. CART and best-ortho-basis: a connection. *Annals of Statistics*, 1997.

Don06. D. Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, 2006.

FPRU10. S. Foucart, A. Pajor, H. Rauhut, and T. Ullrich. The Gelfand widths of ℓ_p-balls for $0 \leq p \leq 1$. *Journal of Complexity*, 2010.

FR13. S. Foucart and H. Rauhut. A Mathematical Introduction to Compressive Sensing. Springer, 2013.

GI10. A. Gilbert and P. Indyk. Sparse recovery using sparse matrices. *Proc. IEEE*, 2010.

HIKP12. H. Hassanieh, P. Indyk, D. Katabi, and E. Price. Nearly optimal sparse Fourier transform. In *Symposium on Theory of Computing*, 2012.

HIS14a. C. Hegde, P. Indyk, and L. Schmidt. Approximation algorithms for model-based compressive sensing. Preprint, available at `http://people.csail.mit.edu/ludwigs/papers/approxmodels.pdf`, 2014.

HIS14b. C. Hegde, P. Indyk, and L. Schmidt. Approximation-tolerant model-based compressive sensing. In *Symposium on Discrete Algorithms (SODA)*, 2014.

HIS14c. C. Hegde, P. Indyk, and L. Schmidt. A fast approximation algorithm for tree-sparse recovery. In *International Symposium on Information Theory (ISIT)*, 2014.

IR13. P. Indyk and I. Razenshteyn. On model-based RIP-1 matrices. In *International Colloquium on Automata, Languages, and Programming*, 2013.

LDP07. M. Lustig, D. Donoho, and J. Pauly. Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic Resonance in Medicine*, 2007.

Mut05. S. Muthukrishnan. Data streams: Algorithms and applications. *Foundations and Trends in Theoretical Computer Science*, 2005.

NPW14. J. Nelson, E. Price, and M. Wootters. New constructions of RIP matrices with fast multiplication and fewer rows. In *Symposium on Discrete Algorithms (SODA)*, 2014.

NT09. D. Needell and J. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Appl. Comput. Harmon. Anal.*, 2009.

RSV08. H. Rauhut, K. Schnass, and P. Vandergheynst. Compressed sensing and redundant dictionaries. *IEEE Trans. Inform. Theory*, 2008.

RV08. M. Rudelson and R. Vershynin. On sparse reconstruction from Fourier and Gaussian measurements. *Comm. Pure Appl. Math.*, 2008.

A Head approximation for the tree-sparsity model

Lemma 1. *Let T be a d-ary tree with $|T| = k$. Moreover, let $\alpha \in \mathbb{N}, \alpha \geq 1$. Then T can be decomposed into a set of disjoint, connected subtrees $S = \{T_1, \ldots, T_\beta\}$ such that $|T_i| \leq d\alpha$ for all $i \in [\beta]$ and $\beta = |S| \leq \lceil \frac{k}{\alpha} \rceil$.*

Proof. We first show that given a d-ary tree U with at least $d\alpha + 1$ nodes, we can find a subtree U' with $\alpha \leq |U'| \leq d\alpha$. Consider the following algorithm FindTree:

```
function FindTree($U$)
    Let $U'$ be the subtree of $U$ maximizing $|U'|$.
    if $|U'| \leq d\alpha$ then
        return $U'$
    else
        return FindTree($U'$)
```

First, note that $|U'| \geq \alpha$ because of the pigeonhole principle and $|U| \geq d\alpha + 1$. Moreover, the size of U decreases with each recursive invocation. Since T is a finite tree, FindTree eventually terminates. When it does, the algorithm returns a subtree U' with $\alpha \leq |U'| \leq d\alpha$.

We use FindTree repeatedly on T in order to build a decomposition S with the desired properties. After identifying a subtree U', we remove it from T and find the next subtree in the remaining tree until at most $d\alpha$ nodes are left. We use this remaining subtree as the final subtree in S.

By construction, S is a set of disjoint, connected subtrees. Moreover, the remaining subtree and each subtree returned by FindTree satisfy $|T_i| \leq d\alpha$. Finally, the decomposition contains at most $\lceil \frac{k}{\alpha} \rceil$ subtrees because every subtree we remove from T has at least α nodes. □

Theorem 4. HeadApprox(x, k, d, p, α) *has a running time of*

$$O(nd^2\alpha + \frac{k}{\alpha}(d\alpha + \log n)(d^2\alpha^2 + \log n)) \ .$$

Proof. Since we invoke ETP with $k' = d\alpha$, the initial call to ETP takes $O(nd^2\alpha)$ time. We can implement the $\arg\max$ in line 5 with a binary heap containing the DP table entries for all nodes and sparsity $d\alpha$. Therefore, this step takes $O(\log n)$ time per iteration of the outer loop.

The cost of the outer loop is dominated by the updates performed in line 8. For each node in $\widehat{\Omega}_i \cup \text{root-path}(\widehat{\Omega}_i)$, we have to update its DP table ($O(d^2\alpha^2)$ time) and then its entry in the binary heap ($O(\log n)$ time). Since $|\widehat{\Omega}_i| \leq d\alpha$ and $|\text{root-path}(\widehat{\Omega}_i)| \leq \lceil \log_d n \rceil$, the total cost over all iterations of the outer loop is $O(\frac{k}{\alpha}(d\alpha + \log n)(d^2\alpha^2 + \log n))$. Combining this with the cost of ETP gives the running time bound in the theorem. □

B Tail approximation for the tree-sparsity model

B.1 Properties of the Pareto curve

The following is an alternative characterization of $s_i(\lambda)$ and $c_i(\lambda)$. The proof follows by an induction over the tree.

Lemma 4. *Let*

$$\Omega_\lambda = \mathop{\arg\max}_{\substack{\Omega \in \mathbb{T}_i,\ \Omega \neq \{\} \\ b_j(\lambda) > 0 \text{ for } j \in \Omega \setminus \{i\}}} \|x_\Omega\|_p^p - \lambda |\Omega| \ .$$

Then

$$\begin{aligned} s_i(\lambda) &= \|x_{\Omega_\lambda}\|_p^p \quad \text{and} \\ c_i(\lambda) &= |\Omega_\lambda| \ . \end{aligned}$$

We now establish a link between $s_i(\lambda)$, $c_i(\lambda)$ and $b_i(\lambda)$.

Lemma 5. *Let $f_i(\lambda) = s_i(\lambda) - \lambda c_i(\lambda)$. Then for $\lambda > \gamma_i$, we have $f_i(\lambda) < 0$. For $\lambda \leq \gamma_i$, we have $f_i(\lambda) = b_i(\lambda)$.*

Proof. We use the characterization of $s_i(\lambda)$ and $c_i(\lambda)$ stated in Lemma 4. For $\lambda < \gamma_i$, we have $i \in \arg\max_{\Omega \in \mathbb{T}_i} \|x_\Omega\|_p^p - \lambda|\Omega|$ and hence

$$\begin{aligned} f_i(\lambda) &= s_i(\lambda) - \lambda c_i(\lambda) \\ &= \|x_{\Omega_\lambda}\|_p^p - \lambda |\Omega_\lambda| \\ &= \max_{\Omega \in \mathbb{T}_i} \|x_\Omega\|_p^p - \lambda |\Omega| \\ &= b_i(\lambda) \end{aligned}$$

Since $f_i(\gamma_i) = 0$, we get $f_i(\lambda) = b_i(\lambda)$ for $\lambda \leq \gamma_i$.

Note that for $\lambda > \gamma_i$, we have $\arg\max_{\Omega \in \mathbb{T}_i} \|x_\Omega\|_p^p - \lambda|\Omega| = \{\}$, and $\{\}$ is the unique maximizer. Since $i \in \Omega_\lambda$ for all values of λ, we get $f_i(\lambda) < b_i(\lambda) = 0$ for $\lambda > \gamma_i$. □

The following lemma shows that for a given value of λ, we can find the next smallest discontinuity in $s_i(\lambda)$ and $c_i(\lambda)$ based solely on the current values of $s_i(\lambda)$ and $c_i(\lambda)$. This is an important ingredient in our algorithm because it allows us to build the Pareto curve incrementally.

Lemma 6. *Let $\lambda \geq 0$ with $\lambda \neq \gamma_i$ for $i \in [n]$ and let*

$$a = \mathop{\arg\max}_{i \in [n],\, b_i(\lambda) = 0} \frac{s_i(\lambda)}{c_i(\lambda)} \ .$$

Then

$$\gamma_a = \max_{i \in [n],\, \gamma_i \leq \lambda} \gamma_i \ .$$

Proof. Since $b_a(\lambda) = 0$, we have $\gamma_a \leq \lambda$ and hence $\gamma_a \leq \max_{i \in [n],\, \gamma_i \leq \lambda} \gamma_i$.

For contradiction, assume that there is a $\lambda \geq \gamma_j > \gamma_a$ and let γ_j be the largest such γ_j. From Lemmas 5 and 2, we have $s_j(\gamma_j) - \gamma_j c_j(\gamma_j) = f_j(\gamma_j) = b_j(\gamma_j) = 0$ and hence $\frac{s_j(\gamma_j)}{c_j(\gamma_j)} = \gamma_j$. Since s_j and c_j are constant in $[\gamma_j, \lambda]$, we have

$$\gamma_j = \frac{s_j(\lambda)}{c_j(\lambda)} \leq \frac{s_a(\lambda)}{c_a(\lambda)} \; . \tag{5}$$

We have $\gamma_j > \gamma_a$ and hence $s_a(\gamma_j) - \gamma_j c_a(\gamma_j) = f_a(\gamma_j) < 0$. Thus, $\gamma_j > \frac{s_a(\gamma_j)}{c_a(\gamma_j)}$. Since s_a and c_a are constant in $[\gamma_j, \lambda]$, we have $\gamma_j > \frac{s_a(\lambda)}{c_a(\lambda)}$, which is a contradiction to (5). □

Let γ_1 and γ_2 be two adjacent discontinuities in $s_i(\lambda)$ and $c_i(\lambda)$ with $\gamma_1 < \gamma_2$. Note that Ω_λ is constant for $\gamma_1 \leq \lambda < \gamma_2$ but $\Omega_{\gamma_2} \neq \Omega_{\gamma_1}$. As our last lemma, we show that Ω_{γ_1} is still an optimal solution for $\lambda = \gamma_2$.

Lemma 7. *Let $\lambda' > 0$ and let*

$$\lim_{\delta \to 0^+} s_i(\lambda' - \delta) = u$$
$$\lim_{\delta \to 0^+} c_i(\lambda' - \delta) = v \; .$$

Then $u - \lambda' v = f_i(\lambda')$.

Proof. First, we show that $f_i(\lambda)$ is continuous. Let

$$b'_i(\lambda) = |x_i|^p - \lambda + \sum_{\substack{j \in \text{children}(i) \\ b_j(\lambda) > 0}} b'_j(\lambda) \; .$$

By definition, we have $b'_i(\lambda) = s_i(\lambda) - \lambda c_i(\lambda) = f_i(\lambda)$. Moreover, an inductive argument similar to the one used for $b_i(\lambda)$ in Lemma 2 shows that $b'_i(\lambda)$ is piecewise linear and continuous. Therefore, $f_i(\lambda)$ is also continuous.

Since $s_i(\lambda)$ and $c_i(\lambda)$ have only finitely many discontinuities, there is an $\lambda'' < \lambda'$ such that for $\lambda'' < \lambda < \lambda'$ we have $s_i(\lambda) = u$ and $c_i(\lambda) = v$. Therefore, we also have $f_i(\lambda) = s_i(\lambda) - \lambda c_i(\lambda) = u - \lambda v$. Moreover, $f_i(\lambda)$ is continuous, so $f_i(\lambda') = \lim_{\delta \to 0} f_i(\lambda' - \delta) = u - \lambda' v$. □

B.2 Finding the Pareto curve

Theorem 5. *Let $s_l^{(i)}$, $c_l^{(i)}$, $active_l^{(i)}$, and $j^{(i)}$ be the values of s_l, c_l, $active_l$, and j after line 8 in iteration i of* FindPareto. *Then $\widehat{\lambda}_i = \max \gamma_l$, where $l \in [n]$ and $\gamma_l < \widehat{\lambda}_{i-1}$. Also, $\widehat{\lambda}_i = \gamma_{j^{(i)}}$. For $\widehat{\lambda}_{i-1} > \lambda \geq \widehat{\lambda}_i$, we have $s_l^{(i)} = s_l(\lambda)$ and $c_l^{(i)} = c_l(\lambda)$. Furthermore, $active_l = true$ if $b_l(\lambda) > 0$ and $active_l = false$ otherwise.*

Proof. We prove the theorem by induction over i. For $i = 1$, the statement of the theorem follows directly from the initialization of the variables in FindPareto.

Now assume that the theorem holds for a given $i > 1$. We need to show that the theorem also holds for $i + 1$.

Since $\widehat{\lambda}_i = \gamma_{j^{(i)}}$, we have $b_{j^{(i)}}(\lambda) > 0$ for $\lambda < \widehat{\lambda}_i$ and hence the update to $\text{active}_{j^{(i)}}$ is correct.

The inner update loop (lines 11 to 17) corresponds directly to the definition of $s_i(\lambda)$ and $c_i(\lambda)$, respectively. Hence we have

$$\begin{aligned} s_l^{(i+1)} &= \lim_{\delta \to 0^+} s_l(\widehat{\lambda}_i - \delta) \\ c_l^{(i+1)} &= \lim_{\delta \to 0^+} c_l(\widehat{\lambda}_i - \delta) \ . \end{aligned}$$

Note that we only have to update the nodes on the path from $j^{(i)}$ to the root because the other nodes are not affected by the discontinuity $\gamma_{j^{(i)}}$.

$s_l(\lambda)$ and $c_l(\lambda)$ are constant up to the next discontinuity given by

$$\gamma' = \max_{\substack{l \in [n] \\ \gamma_l < \widehat{\lambda}_i}} \gamma_l \ .$$

Let $\lambda' = \frac{\widehat{\lambda}_i + \gamma'}{2}$. Applying Lemma 6 with $\lambda = \lambda'$ to line 7 of FindPareto shows that $\gamma' = \widehat{\lambda}_{i+1}$ and $\widehat{\lambda}_{i+1} = \gamma_{j^{(i+1)}}$. □

Theorem 6. *Let $p \geq 1$ and let $x \in R^n$ be the coefficients of a perfect d-ary tree. Then* FindPareto(x, p) *runs in time $O(n \log n)$ for constant d.*

Proof. Since we have a perfect d-ary tree, the depth of any node is bounded by $O(\log n)$. Hence the work of the inner update loop (lines 11 to 17) is bounded by $O(\log n)$ for a single iteration of the outer loop.

We implement the arg max in line 7 with a Fibonacci heap containing the nodes j with $\text{active}_j = \text{false}$. Hence the cost of the arg max is $O(\log n)$ and the cost of the inner update loop remains $O(\log n)$, now in amortized time.

As a result, the total time complexity of all n iterations is $O(n \log n)$. □

B.3 Tail approximation algorithm

Given the shape of the Pareto curve, we want to find the best solution achievable with our extended sparsity budget ck. We implement this search with a single scan over the $\widehat{\lambda}_i$, starting at $\widehat{\lambda}_n$ so that λ is increasing and the corresponding sparsity r_i decreasing. Algorithm 3 contains the pseudo code for this approach.

We first show that FindSolution allows us to reconstruct the support corresponding to a $\widehat{\lambda}_i$ and r_i.

Lemma 8. *Let $x \in \mathbb{R}^n$ be the coefficients corresponding to a d-ary tree and let $p \geq 1$. Then* FindSolution$(x, \widehat{\lambda}_i, p)$ *returns a support $\widehat{\Omega} \in \mathbb{T}_1$ satisfying*

$$\begin{aligned} \left\|x - x_{\widehat{\Omega}}\right\|_p^p + \lambda|\widehat{\Omega}| &= \min_{\substack{\Omega \in \mathbb{T}_1 \\ \Omega \neq \{\}}} \|x - x_\Omega\|_p^p + \lambda|\Omega| \\ \|x_{\widehat{\Omega}}\|_p^p &= s_1(\lambda) \\ |\widehat{\Omega}| &= c_1(\lambda) = r_i \end{aligned}$$

Algorithm 3 (TailApprox) Tail approximation for the tree sparsity model

```
function TailApprox($x, k, c, p$)
    $(\widehat{\lambda}, r) \leftarrow$ FindPareto($x, p$)
    for $i \leftarrow n, \ldots, 1$ do
        if $r_i \leq ck$ then
            return FindSolution($x, \widehat{\lambda}_i, p$)

function FindSolution($x, \lambda, p$)
    CalculateB($1, x, \lambda, p$)
    return $\widehat{\Omega} \leftarrow$ FindSupport($1$)

function CalculateB($i, x, \lambda, p$)
    $\widehat{b}_i \leftarrow |x_i|^p - \lambda$
    for $j \in$ children($i$) do
        CalculateB($j, x, \lambda, p$)
        $\widehat{b}_i \leftarrow \widehat{b}_i + \widehat{b}_j$
    $\widehat{b}_i \leftarrow \max(0, \widehat{b}_i)$

function FindSupport($i$)
    $\Omega_i \leftarrow \{i\}$
    for $j \in$ children($i$) do
        if $\widehat{b}_j > 0$ then
            $\Omega_i \leftarrow \Omega_i \cup$ FindSuppport($j$)
    return $\Omega_i$
```

for $\widehat{\lambda}_{i-1} > \lambda \geq \widehat{\lambda}_i$. Moreover, FindTree *runs in linear time.*

Proof. After the call to CalculateB, we have $\widehat{b}_j = b_j(\lambda_i)$ for $j \in [n]$ (see Lemma 2). Note that FindSupport follows the definition of $s_j(\lambda)$ and $c_j(\lambda)$. Using Lemma 4, we get

$$\widehat{\Omega} = \mathop{\arg\max}_{\substack{\Omega \in \mathbb{T}_1,\ \Omega \neq \{\} \\ b_j(\lambda) > 0 \text{ for } j \in \Omega \setminus \{1\}}} \|x_\Omega\|_p^p - \lambda |\Omega|$$

for $\widehat{\lambda}_{i-1} > \lambda \geq \widehat{\lambda}_i$. Lemma 4 also implies $\|x_{\widehat{\Omega}}\|_p^p = s_1(\lambda)$ and $|\widehat{\Omega}| = c_1(\lambda)$. Applying Theorem 2 then gives $|\widehat{\Omega}| = r_i$.

Negating the above objective function and using $\|x - x_\Omega\|_p^p = \|x\|_p^p - \|x_\Omega\|_p^p$, we get

$$\widehat{\Omega} = \mathop{\arg\min}_{\substack{\Omega \in \mathbb{T}_1,\ \Omega \neq \{\} \\ b_j(\lambda) > 0 \text{ for } j \in \Omega \setminus \{1\}}} \|x - x_\Omega\|_p^p + \lambda |\Omega| \, .$$

Finally, FindSolution makes a constant number of passes over the tree and consequently runs in time $O(n)$. □

We now prove the main result for the tail approximation algorithm.

Theorem 7. *Let $x \in \mathbb{R}^n$ be the coefficients corresponding to a d-ary tree rooted at node 1. Moreover, let $k \geq 1$, $c > 1$ and $p \geq 1$. Then* TailApprox(x, k, c, p) *returns a support $\widehat{\Omega} \in \mathbb{M}^+_{ck}$ satisfying*

$$\left\| x - x_{\widehat{\Omega}} \right\|_p \leq \left(1 + \frac{1}{c-1} \right)^{1/p} \min_{\Omega \in \mathbb{M}_k} \| x - x_\Omega \|_p \ .$$

Furthermore, TailApprox *runs in time $O(n \log n)$.*

Proof. First, note that TailApprox always returns because $ck \geq 1 = r_1$. Moreover, the algorithm only returns if $r_i \leq ck$, so $\widehat{\Omega} \in \mathbb{M}^+_{ck}$ (Lemma 8).

We consider two cases based on $|\widehat{\Omega}|$. If $|\widehat{\Omega}| \geq k$, Lemma 8 implies

$$\begin{aligned} \left\| x - x_{\widehat{\Omega}} \right\|^p_p + \widehat{\lambda}_i |\widehat{\Omega}| &= \min_{\substack{\Omega \in \mathbb{T}_1 \\ \Omega \neq \{\}}} \| x - x_\Omega \|^p_p + \widehat{\lambda}_i |\Omega| \\ &\leq \min_{\Omega \in \mathbb{M}_k} \| x - x_\Omega \|^p_p + \widehat{\lambda}_i |\Omega| \ , \end{aligned}$$

where the last line uses $k \geq 1$. Since $\widehat{\lambda}_i \geq 0$ and $|\widehat{\Omega}| \geq k = |\Omega|$ for $\Omega \in \mathbb{M}_k$, we have

$$\left\| x - x_{\widehat{\Omega}} \right\|_p \leq \min_{\Omega \in \mathbb{M}_k} \| x - x_\Omega \|_p \ .$$

For the case of $|\widehat{\Omega}|$, let i be the final value of the loop counter in TailApprox. In order to establish an approximation guarantee for $\widehat{\Omega}$, we consider the support Ω' corresponding to r_{i+1}. By Theorem 2, this is

$$\Omega' = \mathop{\arg\max}_{\substack{\Omega \in \mathbb{T}_1,\ \Omega \neq \{\} \\ b_j(\lambda) > 0 \text{ for } j \in \Omega \setminus \{1\}}} \| x_\Omega \|^p_p - \lambda |\Omega|$$

for $\widehat{\lambda}_i > \lambda \geq \widehat{\lambda}_{i+1}$. Since the loop in TailApprox continued beyond r_{i+1}, we have $|\Omega'| = r_{i+1} > ck$.

Note that $s_1(\widehat{\lambda}_i) = \| x_{\widehat{\Omega}} \|^p_p$ and $c_1(\widehat{\lambda}_i) = |\widehat{\Omega}|$ (Lemma 8). Moreover, we have

$$\begin{aligned} \lim_{\delta \to 0^+} s_1(\widehat{\lambda}_i - \delta) &= \| x_{\Omega'} \|^p_p \\ \lim_{\delta \to 0^+} c_1(\widehat{\lambda}_i - \delta) &= | x_{\Omega'} | \ . \end{aligned}$$

Using Lemma 7 we get

$$\begin{aligned} \| x_{\Omega'} \|^p_p - \widehat{\lambda}_i |\Omega'| &= f_1(\widehat{\lambda}_i) \\ &= s_1(\widehat{\lambda}_i) - \widehat{\lambda}_i c_1(\widehat{\lambda}_i) \\ &= \| x_{\widehat{\Omega}} \|^p_p - \widehat{\lambda}_i |\widehat{\Omega}| \ . \end{aligned}$$

Equivalently, we have

$$\begin{aligned} \| x - x_{\Omega'} \|^p_p + \widehat{\lambda}_i |\Omega'| &= \left\| x - x_{\widehat{\Omega}} \right\|^p_p + \widehat{\lambda}_i |\widehat{\Omega}| \\ &= \min_{\substack{\Omega \in \mathbb{T}_1 \\ \Omega \neq \{\}}} \| x - x_\Omega \|^p_p + \widehat{\lambda}_i |\Omega| \ , \end{aligned} \tag{6}$$

where the second line follows from Lemma 8. Now let $\Omega^* \in \mathbb{M}_k$ be a support with $\|x - x_{\Omega^*}\|_p = \min_{\Omega \in \mathbb{M}_k} \|x - x_\Omega\|_p$. Since $k \geq 1$, we have

$$\min_{\substack{\Omega \in \mathbb{T}_1 \\ \Omega \neq \{\}}} \|x - x_\Omega\|_p^p + \widehat{\lambda}_i |\Omega| \;\leq\; \|x - x_{\Omega^*}\|_p^p + \widehat{\lambda}_i |\Omega^*| \,. \tag{7}$$

Combining equations (6) and (7), we get

$$\begin{aligned}
\|x - x_{\Omega'}\|_p^p + \widehat{\lambda}_i |\Omega'| &\leq \|x - x_{\Omega^*}\|_p^p + \widehat{\lambda}_i |\Omega^*| \\
\widehat{\lambda}_i (|\Omega'| - |\Omega^*|) &\leq \|x - x_{\Omega^*}\|_p^p - \|x - x_{\Omega'}\|_p^p \\
\widehat{\lambda}_i (ck - k) &\leq \|x - x_{\Omega^*}\|_p^p \\
\widehat{\lambda}_i &\leq \frac{\|x - x_{\Omega^*}\|_p^p}{k(c-1)} \,.
\end{aligned}$$

We combine equations (6) and (7) again, this time for $\widehat{\Omega}$. Moreover, we use our new bound on $\widehat{\lambda}_i$.

$$\begin{aligned}
\left\|x - x_{\widehat{\Omega}}\right\|_p^p + \widehat{\lambda}_i |\widehat{\Omega}| &\leq \|x - x_{\Omega^*}\|_p^p + \widehat{\lambda}_i |\Omega^*| \\
\left\|x - x_{\widehat{\Omega}}\right\|_p^p &\leq \|x - x_{\Omega^*}\|_p^p + \widehat{\lambda}_i k \\
&\leq \|x - x_{\Omega^*}\|_p^p + \frac{\|x - x_{\Omega^*}\|_p^p}{c-1} \\
&\leq \|x - x_{\Omega^*}\|_p^p \left(1 + \frac{1}{c-1}\right) \,.
\end{aligned}$$

Taking the p-th root on both sides gives the guarantee in the theorem.

The running time bound for TailApprox follows directly from the time complexity of FindPareto and FindSolution. □

C Construction of a fast model-RIP matrix

Following the techniques of [NPW14], we demonstrate an easy construction of a matrix that supports fast matrix-vector multiplication, as well as satisfies the model-RIP for the tree-sparsity model. In particular, we prove the following theorem.

Theorem 8. *There exists a randomized construction of $A \in \mathbb{R}^{m \times n}$, with optimal parameters $m = O(k)$, that satisfies the model-RIP for the tree-sparsity model $\mathcal{M}_k$. Moreover, A supports matrix-vector multiplication with complexity $O(n \log n + k^2 \log n \log^2(k \log n))$ for any $k \leq n$. For the regime $k \leq n^{1/2-\mu}$, this complexity can be refined to $O(n \log n)$.*

Proof. We follow a two step-approach to construct A. First, from the results of Rudelson and Vershynin [RV08] as well as the more recent works of [CGV13] and [NPW14], it is known that with high probability, one can construct matrices $F \in \mathbb{R}^{q \times n}$ with $q = O(k \operatorname{polylog} n)$ that satisfy the RIP over *all* sparse vectors in $\mathbb{R}^n$. To the best of our knowledge, the sharpest bounds are achieved by the matrix constructions described in [NPW14], which satisfy the RIP with $q = O(k \log n \log^2(k \log n))$. Their proposed F is of the form SH, where $H \in \mathbb{R}^{n\times n}$ is a Fourier matrix and S is a *sparse* matrix with random ± 1 elements as nonzeros.

For smaller values of k (in particular, for $k \leq n^{1/2-\mu}$ for any $\mu > 0$), an elegant (randomized) approach to construct such an F is described in [AR13]. Specifically, a suitable F can be obtained by concatenating independently chosen linear transformations of the form DH (where $H \in \mathbb{R}^{n\times n}$ is a Fourier or Hadamard matrix and $D \in \mathbb{R}^{n\times n}$ is a diagonal matrix with random ± 1 elements along the diagonal), followed by left multiplication with any row-orthonormal matrix (such as a row-selection matrix) of size $q \times n$, where $q = O(k \log n)$.

In either case, F provides a stable embedding of the set of all k-sparse signals into $\mathbb{R}^q$ with high probability. In other words, given any subset of indices $\Lambda \subset [n]$ with cardinality k, the following relation holds for all vectors x supported on Λ:

$$(1-\delta_F)\|x\|_2^2 \leq \|Fx\|_2^2 \leq (1+\delta_F)\|x\|_2^2$$

for some small constant δ_F.

Next, consider a random matrix $G \in \mathbb{R}^{m\times q}$ that satisfies the following *concentration-of-measure* property: for any $v \in \mathbb{R}^q$, the following holds:

$$\mathbb{P}(|\|Gv\|_2^2 - \|v\|_2^2| \geq \epsilon\|v\|_2^2) \leq 2e^{-c\frac{n}{2}\varepsilon^2}, \quad \forall\, \varepsilon \in (0, 1/3)\ . \tag{8}$$

Again, it is known that a matrix $G = \frac{1}{\sqrt{m}}\bar{G}$, with the elements of $\bar{G} \in \mathbb{R}^{m\times q}$ drawn from a standard normal distribution, satisfy (8). Now, choose any index set $\Lambda \in \mathbb{M}_k$ belonging to the tree-sparsity model, and a small constant $\delta_G > 0$. From Lemma 2.1 of [RSV08], the following property holds for all x supported on Λ: if $\delta := \delta_F + \delta_G + \delta_F\delta_G$, then

$$(1-\delta)\|x\|_2^2 \leq \|GFx\|_2^2 \leq (1+\delta)\|x\|_2^2$$

with probability exceeding

$$1 - 2\left(1 + \frac{12}{\delta_G}\right)^k e^{-\frac{c}{9}\delta_G^2 m}\ .$$

In other words, for signals with a given support set $\Lambda \in \mathbb{M}_k$, the probability that GF fails to have a isometry constant δ is no greater than $2\left(1 + \frac{12}{\delta_G}\right)^k e^{-\frac{c}{9}\delta_G^2 m}$. The total number of supports Λ in the tree-sparsity model can be upper bounded by $(2e)^k/(k+1)$ [BCDH10]. Therefore, performing a union bound over all possible Λ, the probablity that GF fails to have an isometry constant δ over the model $\mathcal{M}_k$ is upper bounded by

$$2\frac{(2e)^k}{k+1}\left(1 + \frac{12}{\delta_G}\right)^k e^{-\frac{c}{9}\delta_G^2 m}\ . \tag{9}$$

Choosing $m = O(k)$ and δ_G sufficiently small, (9) can be made exponentially small. Therefore, with high probability, $A = G \cdot F$ satisfies the RIP over all signals belonging to the model $\mathcal{M}_k$, with $m = O(k)$ and a sufficiently small constant δ.

Multiplication of F with any vector $x \in \mathbb{R}^n$ incurs $O(n \log n)$ complexity, while multiplication of G with Fx incurs a complexity of $O(k \times q)$. Therefore, the overall complexity scales as $O(n \log n + kq)$. Substituting for the best available choices of F for different ranges of k, we obtain the stated result. $\square$